

# A probabilistic methodology for multilabel classification

Alfonso E. Romero<sup>1</sup>, Luis M. de Campos<sup>2</sup>

<sup>1</sup>Centre for Systems and Synthetic Biology, and  
Department of Computer Science,  
Royal Holloway, University of London,  
Egham, TW20 0EX, United Kingdom  
`aeromero@cs.rhul.ac.uk`

<sup>2</sup>Departamento de Ciencias de la Computación e Inteligencia Artificial  
E.T.S.I. Informática y de Telecomunicación, CITIC-UGR  
Universidad de Granada, 18071 – Granada, Spain  
`lci@decsai.ugr.es`

## Abstract

Multilabel classification is a relatively recent subfield of machine learning. Unlike to the classical approach, where patterns are labeled with only one category, in multilabel classification, an arbitrary number of categories is chosen to label a pattern. Due to the problem complexity (the solution is one among an exponential number of alternatives), a very common solution (the *binary* method) is frequently used, learning a binary classifier for every category, and combining them all afterwards. The assumption taken in this solution is not realistic, and in this work we give examples where the decisions for all the labels are not taken independently, and thus, a supervised approach should learn those existing relationships among categories to make a better classification. Therefore, we show here a generic methodology that can improve the results obtained by a set of independent probabilistic binary classifiers, by using a combination procedure with a classifier trained on the co-occurrences of the labels. We show an exhaustive experimentation in three different standard corpora of labeled documents (Reuters-21578, Ohsumed-23 and RCV1), which present noticeable improvements in all of them, when using our methodology, in three probabilistic base classifiers.

**Keywords:** multilabel classification, probabilistic classifiers, text classification

## 1 Introduction

In this work we present a novel solution for the multilabel categorization problem. In this kind of problems, a subset of categories (instead of just one) is

assigned to each pattern. Multilabel classification problems arise, in a natural way, almost completely in information processing, concretely on the subfield of automatic document categorization [21]. Due to their nature, an important number of text corpora are of a multilabeled kind. For example, news articles can often belong to more than one category (this is the case of the Reuters-21578 [11] and the RCV1 [12] collections, which are composed of articles from the Reuters agency). In other domains, multiple labels are assigned as metadata which give a better description of the documents (this occurs, for example, in scientific papers which have associated keywords from a controlled vocabulary, like the Mathematics Subject Classification, or the MeSH Thesaurus). On the other hand, multilabel patterns occur very commonly in the Internet: in many blog applications, blog posts for example, can be categorized with an arbitrary number of labels. Furthermore, in collaborative environments (like folksonomies [25]) where users can add tags, multilabel is an ordinary process. Due to this fact, sometimes the word “tag” or “label” is used instead of “category”, they are all synonyms.

More recently, multilabel classification has been useful for different domains like, for instance, analysis of musical emotions [22], scene or image categorization [6] or protein function prediction [16].

Although each pattern has a determined set of assigned labels, and they are given as a whole, the normal approach to solve this problem is just ignore this fact and concentrate in obtaining good solutions to the individual binary problems (i.e., deciding for each label if it should be assigned to the pattern or not). Here, we propose a solution which takes into account the inter-category dependence, trying to find natural associations in order to improve the final categorization results.

The content of this paper is organized as follows: first of all (in section 1.1) we recall the well-known problem of multilabel supervised categorization, reviewing some previous works in this area (section 1.2) together with a brief explanation (section 1.3) of what is the semantic of adding multiple labels to a pattern instead of one. Based on probabilistic foundations, our approach is presented in section 2, which results in two different models. An extensive experimentation with test collections coming from the text categorization field will be carried out in section 3 to prove the scope of our proposal. Finally, in the light of the results previously obtained, several conclusions and future works will be pointed out in section 4.

## 1.1 Multilabel supervised categorization

The problem of supervised multilabel classification (see, for instance [23]) deals with supervised learning, where the associated labels can be a set of undetermined size. Formally, it can be stated as follows: given a set of categories  $\mathcal{C} = \{c_1, \dots, c_p\}$ , an input pattern space  $\mathcal{X}$ , and a set of labeled data composed of patterns and the set of assigned labels,  $\{x_i, y_i\}_{i=1, \dots, n}$  where  $y_i \subset 2^{\mathcal{C}} \setminus \emptyset$ ,  $x_i \in \mathcal{X}$ , learning a multilabel classifier means inferring a function  $f : \mathcal{X} \longrightarrow 2^{\mathcal{C}} \setminus \emptyset$  (in other words, a function able to assign non-empty subsets of labels to any unlabeled pattern).

In order to cope with this exponential output space, the classical approach consists in dividing the problem into  $|\mathcal{C}|$  binary independent problems, and therefore learning  $|\mathcal{C}|$  binary classifiers  $f_i : \mathcal{X} \longrightarrow \{c_i, \overline{c_i}\}$  (which decide whether

the category is given or not to the pattern). Although this is a naive solution, it works reasonably well, and in multilabel classification literature this can be used as a baseline. In this work we shall propose a more advanced solution to this problem, trying to capture relationships among categories, and then, improving classification results.

## 1.2 Related work

There is more than one hundred references partially related with multilabel categorization, most of them in the last years<sup>1</sup>. At first sight, the two possible solutions to this problem basically consist of transforming it to a single-label one, or adapting a learning procedure to work with these multiple labels at the same time. This taxonomy is given in [23], naming the former solutions *problem transformation methods* and *algorithm adaptation* the latter ones. Because our contribution adapts a learning procedure to multilabel learning, we review here only approaches of the second kind.

Almost all the typical classification algorithms have a multilabel version. For example, an adaptation of the entropy formula of the C4.5 tree learning algorithm has been proposed in [1] for multilabel classification. In the lazy algorithms field, variations of the  $k$ -NN algorithm have been presented for this kind of problems [13, 29]. Also, in [3] a  $k$ -NN is combined with a logistic regression classifier (in a different way that we do) to cope with multiple labels. Of course, variations on the SVM algorithm are shown in [8] where both intra-class dependencies and a improvement of the definition of margin for multilabel classification are used to build a new model.

On the other hand, there are methods dealing within the probabilistic framework and therefore closer to our approach. In [15], a generative model is trained using training data, and completed with the EM algorithm, and computed the most probable vector of categories that should be assigned to the document. A subset of Reuters-21578 is used for experimentation, and noticeable improvements are shown.

A generative model is also presented in [24]. Here, the main assumption is that words in documents belonging to several categories can be characterized as a mixture of characteristic words related to each of the categories, being this assumption confirmed with experimentation. Both first (PMM1) and a second (PMM2) order models are built, and learning algorithms (using a MAP estimation) are proposed to both alternatives. Experiments are carried out with webpages gathered from the `yahoo.com` server. Presented results are good, and improve other methods as SVM, naive Bayes and  $k$ -NN.

More recently [4] proposed a novel method, called probabilistic classifier chains (generalizing the classifier chains) which exploits label dependence showing that the method outperform others in term of loss functions. This is claimed via an extensive experimentation with artificial and real datasets.

## 1.3 The semantic of assigning multiple labels

The question asked in this section is whether we can enumerate all the casuistic of assigning multiple labels. We present three common non-independent phenomena with examples that may occur easily in this kind of problems:

---

<sup>1</sup>See <http://www.citeulike.org/group/7105/tag/multilabel>, and references therein.

1. *Mixture of topics.* A pattern matches all the abstract description of several categories. This is the case, for example of a medical paper which deals with several topics represented as MeSH keywords.
2. *Contextualization.* Some tags are added in order to precise the context in which other label is used. For example, in scene classification, a picture of fishermen working in the coast of Motril tagged with “sea” and “people” can be contextualized with “town” in order to distinguish it from submarine photos.
3. *Non overlapping labels.* Some subsets of labels do not admit patterns belonging to all of them. For example, in music classification, it is inconceivable to have songs tagged with both “baroque” and “reggae”, although other combinations as “flamenco” and “jazz” are possible.

The only “pure” multilabel phenomenon is the first one. The second and the third denote that the occurrence of labels is not only based on the content of the pattern, but also in the occurrence (or not) of other labels. For the second case, a label is added to contextualize two previously given labels. That is to say, the occurrence of a certain subset of labels increases the likelihood of other being added. On the other hand, in the third case, a song labeled with “flamenco” (at a certain degree) and “reggae” (in a lower degree) may be detected by a classifier as an “anomaly”, and be labeled only with “flamenco” (given that the system has previously learned that the label “flamenco” gives information about the low likelihood of using the label “reggae” having the first).

Although the first phenomenon can be initially captured by different binary classifiers, the second and the third can be precised by only looking at the labels of a training set<sup>2</sup>, and not to its content. Therefore, our contribution will be to state that the final labeling of a pattern, in a certain category, will be a combination of the binary classifier (with the evidence given in the other categories by the other binary classifiers) taking into account the existing relationships among categories captured in the training set. In fact, this issue of label dependence have been recently shown to be crucial in multilabel learning [5]. All of this will be modeled in a probabilistic framework, which will give us a rich language to describe this procedure.

## 2 A probabilistic model for multilabel classification

### 2.1 Set up

Let  $x_i \in \mathcal{X}$  be a pattern. Suppose we have a set of  $p$  categories  $\mathcal{C} = \{c_1, \dots, c_p\}$ . For every category  $c_j$ , we define a binary random variable  $C_j = \{c_j, \bar{c}_j\}$ . Let us assume we have  $p$  probabilistic classifiers, based only on the content of the patterns. Thus, the conditional probability  $p_j(c_j|x_i)$  represents the probability that the pattern  $x_i$  is labeled with  $c_j$  (the subindex  $j$  indicates that this distribution is different and independent for every category). Assuming we have a perfect knowledge of the underlying probability distribution, these classifiers

---

<sup>2</sup>Or adding expert knowledge with explicit relations among the labels, like a hierarchy.

define  $p$  labeling rules as follows (Bayes optimal classifier): “classify  $x_i$  as  $c_j$  if  $p_j(c_j|x_i) > p_j(\bar{c}_j|x_i)$ ” or, alternatively “classify  $x_i$  as  $c_j$  if  $p_j(c_j|x_i) > 0.5$ ”.

For every category  $c_j$ , we shall also define a random vector of binary variables  $\mathbf{L}_j = (l_{j1}, \dots, l_{jp-1})$ , which represents the labeling of a pattern with the other  $p-1$  classifiers. We shall note for  $\mathbf{l}_j$  a particular value of the vector  $\mathbf{L}_j$  (that is to say, a label for the pattern in all the categories except the  $j$ -th one). Thus, every component of the vector  $l_{jk}$  is binary, and corresponds to the variable  $C_k$  if  $k < j$  and to  $C_{k+1}$  otherwise.

Our aim is to give a model which describes the probability of a class given knowledge of both the content of the pattern,  $x_i$ , and the labeling of the other categories ( $\mathbf{l}_j$ ). In other words, an expression for the probability  $p_j(c_j|x_i, \mathbf{l}_j)$ .

## 2.2 The proposed model

Our model will start taking a reasonable simplifying assumption (*general naive Bayes assumption*<sup>3</sup>): given the true value of a category  $j$  for a pattern, the probabilities of finding a certain pattern  $x_i$  and a certain labeling on the other categories,  $\mathbf{l}_j$ , are independent, that is

$$p_j(x_i, \mathbf{l}_j|c_j) = p_j(x_i|c_j) p_j(\mathbf{l}_j|c_j), \quad \forall j \in \{1, \dots, p\}.$$

Then, using Bayes’ theorem with this assumption, we compute the desired probability:

$$\begin{aligned} p_j(c_j|x_i, \mathbf{l}_j) &= \frac{p_j(x_i, \mathbf{l}_j|c_j) p_j(c_j)}{p_j(x_i, \mathbf{l}_j)} = \frac{p_j(x_i|c_j) p_j(\mathbf{l}_j|c_j) p_j(c_j)}{p_j(x_i, \mathbf{l}_j)} \\ &= \frac{p_j(c_j|x_i) p_j(x_i) p_j(c_j|\mathbf{l}_j) p_j(\mathbf{l}_j)}{p_j(c_j) p_j(x_i, \mathbf{l}_j)} \\ &= \left( \frac{p_j(x_i) p_j(\mathbf{l}_j)}{p_j(x_i, \mathbf{l}_j)} \right) \left( \frac{p_j(c_j|x_i) p_j(c_j|\mathbf{l}_j)}{p_j(c_j)} \right). \end{aligned}$$

The first term is a proportionality factor which does not depend on the category. Therefore we get the expression,

$$p_j(c_j|x_i, \mathbf{l}_j) \propto \frac{p_j(c_j|x_i) p_j(c_j|\mathbf{l}_j)}{p_j(c_j)},$$

which leads us to the final formula:

$$p_j(c_j|x_i, \mathbf{l}_j) = \frac{p_j(c_j|x_i) p_j(c_j|\mathbf{l}_j)/p_j(c_j)}{p_j(c_j|x_i) p_j(c_j|\mathbf{l}_j)/p_j(c_j) + p_j(\bar{c}_j|x_i) p_j(\bar{c}_j|\mathbf{l}_j)/p_j(\bar{c}_j)} \quad (1)$$

Taking into account we are in binary classification, it holds that  $p_j(\bar{c}_j) = 1 - p_j(c_j)$ ,  $p_j(\bar{c}_j|x_i) = 1 - p_j(c_j|x_i)$  and  $p_j(\bar{c}_j|\mathbf{l}_j) = 1 - p_j(c_j|\mathbf{l}_j)$ . The values  $p_j(c_j|x_i)$

<sup>3</sup>The term *general* is used here in a sense of analogy with the “classic” naive Bayes assumption (given the true category of a pattern, the joint probability of its features factorizes as a product of independent probability distributions). The previous labelings of the document along with its content can be considered as “general” features, and this assumption means that we apply it to that set of features.

can be simply obtained with any probabilistic binary classifier for the category  $c_j$ . Prior probabilities  $p_j(c_j)$  are estimated as the number of patterns which belong to class  $c_j$  over the number of all patterns. Probabilities  $p_j(c_j|\mathbf{l}_j)$  can be estimated, through a learning process, from the labels of the training data, where every pattern has as features some binary values telling if the pattern belong or not to any of the  $p - 1$  categories (all categories except  $j$ ). So, in our model, we need to train  $p$  binary classifiers for the content of the pattern, and  $p$  binary classifiers in the labels assigned to patterns in the training set.

A last comment should be clarified in the model. Given a pattern  $x_i$  to classify, we easily compute, for a category  $c_j$  both the values  $p_j(c_j)$  and  $p_j(c_j|x_i)$ . However, to obtain the probability  $p_j(c_j|\mathbf{l}_j)$  we need to know the real assignments of the labels (0 or 1) which are not  $c_j$ . In our model, we shall make a second assumption, approximating  $\mathbf{l}_j$  for  $\hat{\mathbf{l}}_j$ , which is computed as follows:

$$\hat{\mathbf{l}}_j = (\tau(p_k(c_k|x_i)))_{k \in \{1, \dots, p\} \setminus j}. \quad (2)$$

Where  $\tau$  is a threshold function ( $\tau(z) = 0$  if  $z < 0.5$ ,  $\tau(z) = 1$  otherwise). In other words,  $\hat{l}_{jk} = \arg \max_{c_k} p_k(c_k|x)$ . Therefore  $p_j(c_j|\mathbf{l}_j)$  will be approximated by  $p_j(c_j|\hat{\mathbf{l}}_j)$ .

### 2.3 An improved version of the model

Note that the model summed up in eq. (1) does not really need the assumption that the random vectors  $\mathbf{L}_j$  are made of binary variables. In fact, the only required binary variables are the  $C_j$  ones. Thus, the same equation is equally true if the variables in  $\mathbf{L}_j$  are continuous.

Besides, we can rewrite a different approximation of  $\mathbf{l}_j$  than the one given in eq. (2) by removing the threshold function:

$$\hat{\mathbf{l}}_j = (p_k(c_k|x_i))_{k \in \{1, \dots, p\} \setminus j}. \quad (3)$$

This means that the components of the vector  $\hat{\mathbf{l}}_j$  are values in  $[0, 1]$  which represent our degree of belief of that label being assigned to the pattern. Note that, in order to use this extended version of the model we need a classifier to compute  $p_j(c_j|\mathbf{l}_j)$  which is capable of dealing with continuous inputs (in the previous version we could assume the inputs were binary, and so the classifier).

### 2.4 The algorithm

We present finally the proposed algorithm for multilabel classification. We need, as input data, the prior probabilities of each category ( $p_j(c_j)$ ),  $p$  content-only binary classifiers (capable of giving an output  $p_j(c_j|x_i)$ ), and  $p$  classifiers which predict the category  $j$  on the values of the labels different than  $j$ :  $p_j(c_j|\mathbf{l}_j)$ .

1. Given a pattern  $x_i$ , we obtain the values  $p_j(c_j|x_i)$  from  $p$  binary probabilistic classifiers ( $j = 1, \dots, p$ ).
2. For every category  $c_j, j \in \{1, \dots, p\}$ ,
  - (a) Using  $p_k(c_k|x_i)$ , we compute  $\hat{\mathbf{l}}_j$  either by eq. (2) or eq. (3). This will be used as an approximation of  $\mathbf{l}_j$ .

- (b) We compute the likelihood of the category given the probabilities of the other categories as  $p_j(c_j|\mathbf{l}_j)$ .
  - (c) We compute the probability value  $p_j(c_j|x_i, \mathbf{l}_j)$ , following eq. (1).
3. The scores of the pattern  $x_i$  in the set of categories  $\mathcal{C}$  are the values  $p_j(c_j|x_i, \mathbf{l}_j)$ . They can be thresholded, optionally, if we are doing hard categorization.

The whole process is summed up in Figure 1.

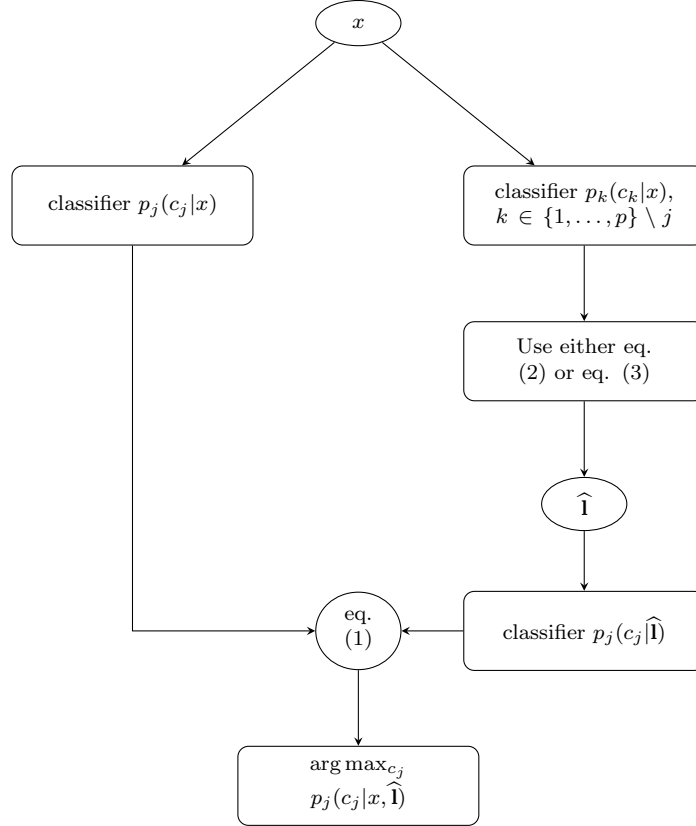


Figure 1: The general algorithm.

## 2.5 Some additional comments

We have shown a new method to deal with relationships among labels, using probabilistic classifiers. This is why it is presented as a “methodology”. For a concrete problem, two decisions should be made about which underlying models should be used: first for the content classifiers, and second for the label classifiers. This election relies heavily on the kind of problem selected, and is suitable of previous experimentation to find a working set of methods. Also one may note that both elections are independent (so, with a set of  $n_c$  probabilistic content classifiers and  $n_l$  label classifiers,  $n_c n_l$  combinations are possible, each one with two possibilities for computing the vectors  $\mathbf{l}_j$ ).

We shall expose in next section an experimentation to test the validity of this approach, where some combinations of classifiers will be selected following a certain criterion. Of course, we do not aim being exhaustive and giving a long list of experiments. We are aware that many different collections and combinations of classifiers could be selected, so we tried to make a good experimentation by selecting a restricted but representative set of classifiers.

## 3 Experimentation

### 3.1 Corpora

Three different document categorization corpora have been used for experimentation. We describe them now, together with the preprocessing procedure used to obtain the term vectors.

First of all Reuters-21578 (see, for instance [11]) is a collection of 21578 news articles. We have used the most famous split (the one named ModApte), which divides the set of documents into a training and a test set, and categories only assigned to documents in the test set are removed, (then resulting only 90 of them).

The Ohsumed collection [9] is a set of 348566 references from MEDLINE, an online medical information database. For every record the assigned MeSH terms (categories) are given. Because the number of categories of the MeSH thesaurus is huge, it is often chosen a subset of 23 categories (*heart diseases*), which are the root of some categories in a hierarchy. Documents which do not belong to that subtree of categories, are discarded, and the resulting corpus is called Ohsumed-23. This is the methodology followed in [10].

Finally, the RCV1 corpus is a relatively more recent corpus (see [12]), also based on Reuters news stories. It contains many documents (806791 for the final version, RCV1-v2), where the documents are preprocessed, with stop-words removed, and terms already stemmed. The number of categories named “topic codes”<sup>4</sup> is 103 (after the split). An standard split is also provided, called LYRL2004, which gives a training set with over 23000 documents, and a test set with 781000. We have removed two categories which appear only in the training set (reducing the number to 101).

In the first two cases, the stopwords list used consists of 571 stopwords of the SMART retrieval system [20]. Also, the English stemming algorithm of the Snowball package [18] was applied to resulting words. In the Reuters-21578 also XML marks were removed.

In order to reduce the size of the lexicon, terms occurring in less than three documents were removed in Reuters-21578 and Ohsumed-23 (following the guidelines of [10]), and in less than five documents in RCV1, as done in [12]. All document preprocessing stage was made with DauroLab [19].

### 3.2 Evaluation measures

We have made in all cases hard categorization (assigning or not every label) and then, we have used suitable measures for this task. Being defined for every

---

<sup>4</sup>Other two disjoint set of categories, “industry codes” and “region codes” are given for this corpus, but they are not normally used for categorization.



category  $c_i$  the precision and recall ( $\pi_i$  and  $\rho_i$ , respectively) as:

$$\pi_i = \frac{TP_i}{TP_i + FP_i}, \quad \rho_i = \frac{TP_i}{TP_i + FN_i},$$

where  $TP_i$ ,  $FP_i$  and  $FN_i$  stand for “true positives”, “false positives” and “false negatives” of the  $i$ -th category. We have therefore selected  $F_1$  measure adapted for categorization (the harmonic mean between precision and recall as previously defined), in its macro and micro averaged versions (denoted by  $MF_1$  and  $\mu F_1$ , respectively). See [21] for more details.

All the probabilistic classifiers have a natural threshold in 0.5. We have not made any threshold tuning because it was not the aim of this paper. Anyway, it could be made independently, likely improving the results (see the discussion in section 4).

### 3.3 Basic probabilistic classifiers

In order to have, first a baseline, and secondly a basic content classifier to be used afterwards for our model, we propose three different classifiers, widely used in the literature. One which usually obtains discrete results, and two with better results, all of them of a different nature. For the first case, we selected the multinomial naive Bayes, in its binary version [14]. For the second case, a  $k$ -NN classifier has been used, normalizing the output in order to obtain a value in  $[0, 1]$  which can be interpreted as a probability. Finally, a linear SVM, with probabilistic output<sup>5</sup>, as performed by Platt’s algorithm [17] was chosen. We recall that the flexibility of this procedure is very high because any probabilistic classifier could be used instead of these three.

For the  $k$ -NN classifier, the best performing value of  $k$  was selected, based on previous experimentation existing in other works. Thus, we chose  $k = 30$  for Reuters and  $k = 45$  for Ohsumed-23 (as set in [10]). For RCV1, a value of  $k = 100$  was used [12]. Also, following those references, we have performed feature selection in Reuters (1000 features by information gain, using the “sum” combination [21]), and in RCV1 (8000 features by  $\chi^2$ , using the “max” combination). No feature selection was performed in Ohsumed-23 as noted in [10].

The implementation used for the algorithms was that contained in the DaurLab [19] package, except for the SVM where libsvm [2] was chosen.

### 3.4 Label classifier

Based on previous experimentations (not shown here), we have chosen a logistic regression-based classifier to model the posterior probability distribution of the category given the correct labels of all the other categories, namely:

$$p_j(c_j|\mathbf{l}_j) = \frac{1}{1 + e^{-(w_0 + \mathbf{w} \cdot \mathbf{l}_j)}},$$

where  $w_0$  is a real scalar and  $\mathbf{w}$  a  $p - 1$  dimensional real vector, and “ $\cdot$ ” is the usual dot product. The parameters of this model are learned to maximize a certain criterion (generally a maximum likelihood approach).

We have selected this classifier instead of other proposals for three reasons: first of all, it can deal with real inputs. Second, it is a discriminative method,

---

<sup>5</sup>The one implemented in LibSVM.

which does not try to find the joint distribution of  $C_j$  and  $L_j$  (in which we are not interested). Finally, it is very simple, fast to learn, and works reasonably well in almost all the environments.

In order to have accurate estimates, and because the dimensionality of the problem  $p$  is high, the method selected to learn the weights is a Bayesian logistic regression, concretely the one proposed in [7], with Gaussian priors. The implementation used here is the one included with the WEKA package, with the default parameters (see [26] for details).

One of the flexibilities of any logistic regression classifier is that it can model distributions with real inputs. As we have two approximations for the input vector in this classifier, we can propose different models, which will be called M1 (corresponding to eq. (2)) and M2 (for equation (3)). It seems reasonable that, if the content classifier performs well, the model M2 will be more accurate than the model M1. We shall discuss this point afterwards.

### 3.5 Results

We presents the results for Reuters-21578 (in Table 1), Ohsumed-23 (in Table 2) and RCV1 (in Table 3). NB denotes the multinomial naive Bayes,  $k$ -NN the  $k$  nearest neighbors classifier, and SVM the linear support vector machine with probabilistic outputs. A classifier + M1 or M2 notes our proposal with the binary or real inputs presented to the logistic regression, as explained before. Also, the results of the base classifier have been shown for comparison purposes.

We give the micro and macro  $F_1$  values ( $\mu F_1$  and  $MF_1$ , respectively), and the percentage of improvement (positive or negative) over the baseline, noted by  $\Delta$ .

In almost all of the cases our proposal with the M2 version of the algorithm improved both measures from the baseline. On the other hand, the M1 version presented not so systematic improvements. The results were noticeable, even reaching a gain of 72% in the case of the macro  $F_1$  measure with respect to the baseline (which is a remarkable achievement). We will discuss the results later in section 4.

Besides, we have run statistical significance tests for every couple of classifiers (a basic probabilistic classifier, and our proposal, either with the M1 or the M2 configuration). For the micro measure, a micro sign s-test was performed, and for the macro measure, we chose the macro S-test. Both are presented in [27] and constitute the nowadays standard for comparing this kind of experiments. Nevertheless it should be noticed that the s-test is not specifically designed for the  $F_1$  measure, taking into account both true positives and true negatives (as shown in section 3.2  $F_1$  only considers true positives). Also, note that the macro S-test does not take into account the amount of improvement, but the number of categories where the measure is improved, leading sometimes to counter-intuitive results (with a non-significant high improvement in macro due to the improvement of only very few categories).

In the test the first system,  $A$ , will be the best performing one, in terms of micro or macro  $F_1$  measure, being  $B$  the second. The null hypothesis is that  $A$  is similar to  $B$ . On the other hand, the alternative hypothesis says that  $A$  is better than  $B$ .

If the p-value is less than 0.01, we show in the tables the sign  $\ll$  (resp.  $\gg$ ) to note that our baseline plus M1 or M2 is significantly better (resp. worse) than

Model	$\mu F_1$	$\Delta$	s-test	$MF_1$	$\Delta$	S-test
NB	0.61765	–	–	0.25145	–	–
NB + M1	0.65697	+6.36%	$\ll$	0.29183	+16.06%	$\ll$
NB + M2	0.65830	+6.58%	$\ll$	0.29293	+16.49%	$\ll$
k-NN	0.78131	–	–	0.27024	–	–
k-NN + M1	0.65615	-16.02%	$\gg$	0.35023	+29.6%	$<$
k-NN + M2	0.77718	-0.53%	$\gg$	0.40518	+49.93%	$\ll$
SVM	0.87102	–	–	0.30722	–	–
SVM + M1	0.85032	-2.38%	$\gg$	0.35892	+16.83%	$<$
SVM + M2	0.87970	+1.00%	$\sim$	0.36592	+19.11%	$\ll$

Table 1: Results for Reuters-21578 corpus

Model	$\mu F_1$	$\Delta$	s-test	$MF_1$	$\Delta$	S-test
NB	0.56166	–	–	0.49985	–	–
NB + M1	0.57789	+2.89%	$\ll$	0.52204	+4.44%	$\ll$
NB + M2	0.57832	+2.97%	$\ll$	0.52208	+4.45%	$\ll$
k-NN	0.43487	–	–	0.29451	–	–
k-NN + M1	0.54352	+24.98%	$\sim$	0.50804	+72.50%	$<$
k-NN + M2	0.47816	+9.95%	$\ll$	0.35945	+22.05%	$\ll$
SVM	0.64802	–	–	0.59596	–	–
SVM + M1	0.66669	+2.88%	$\sim$	0.62482	+4.84%	$\ll$
SVM + M2	0.65871	+1.65%	$\sim$	0.61175	+2.65%	$\ll$

Table 2: Results for Ohsumed-23 corpus

the baseline alone. The signs  $<$  and  $>$  are used to indicate the same fact if the p-value is between 0.01 and 0.05. With the sign  $\sim$  we point out no significant difference between the two systems despite the results in the measure.

The improvement for the micro measure in the RCV1 corpus were small but it should be noted that the baseline value (0.80571) was very close to the best result obtained by Lewis [12] with SVM (0.816) and a sophisticated threshold tuning algorithm (ScutFBR.1). Perhaps better results than this high value are very difficult to get.

## 4 Conclusions and future works

Once shown the results, we may state that the methodology presented is useful for improving classification results in a multilabeled environment. Concretely, in the presented tables, the following facts are found:

- The use of this technique clearly improves the classification results of the baseline. For the macro experiments, all the measures are improved from the baselines. In the micro ones, also good improvements are found, especially for the M2 version of the classifier, which improves the baseline in all but one cases (Reuters with the  $k$ -NN where the loss is around 0.5%).
- Comparing both approaches, the model M1 (binarized) performs, in general, worse than the continuous (M2) version. This is because the binariza-

Model	$\mu F_1$	$\Delta$	s-test	$MF_1$	$\Delta$	S-test
NB	0.61828	–	–	0.27759	–	–
NB + M1	0.64287	+3.98%	$\ll$	0.29124	+4.92%	$\ll$
NB + M2	0.64379	+4.13%	$\ll$	0.29160	+5.05%	$\ll$
k-NN	0.68698	–	–	0.27494	–	–
k-NN + M1	0.60402	-12.08%	$\gg$	0.40528	+47.40%	$\ll$
k-NN + M2	0.72488	+5.52%	$\sim$	0.40404	+46.96%	$\ll$
SVM	0.80571	–	–	0.36563	–	–
SVM + M1	0.78306	-2.81%	$\gg$	0.40280	+10.17%	$\sim$
SVM + M2	0.80769	+0.24%	$\sim$	0.40672	+11.23595%	$\ll$

Table 3: Results for RCV1 corpus

tion procedure removes some information represented in the granularity of the assignment that is well captured by the classifier.

- In general, we can say that this methodology seems to benefit classification of less populated categories (those with a very low prior) a lot, without harming more frequent categories. This is because, in general, the macro measures (where all categories weight the same) are heavily lifted up, and micro measures are improved in a less degree.

The combination of this method with any of the well-known thresholding techniques [28] is an open question. The first and obvious question is: what happens if a thresholding algorithm is applied after our method (instead of just binarizing the final probability output)? At a first glance it should improve the results, but how? Is this combination worthwhile? Also, a more sophisticated variant of the M1 method, with a better thresholding function  $\tau$  (using learned values for the thresholds, not just 0.5) can be studied. Or even, a combination of both proposals. As future works we would like to study the synergies between our proposal for improving multilabel classification and some thresholding techniques.

Another open question is the following: using the same approach, a different independence assumption than the one given in eq. (1) could be given, leading to other models. In particular, we would like to explore methods where the independence assumption is relaxed, because we think that a complete independence may be unrealistic in some categories.

Finally, we would like to use this methodology in a different environment. We have selected document categorization for validating this model because it is a natural form of multilabeled patterns. In the future, we would like to work with different datasets as, for example, musical patterns, protein data or social network data, which we think are also suitable for this method.

**Acknowledgments** This study has been jointly supported by the Spanish *Ministerio de Ciencia e Innovación*, under the research programme *Consolider Ingenio 2010*, and the *Consejería de Innovación, Ciencia y Empresa de la Junta de Andalucía* in the projects MIPRCV (CSD2007-00018), and P09-TIC-4526, respectively.

## References

- [1] A. Clare, and R. D. King, *Knowledge Discovery in Multi-label Phenotype Data*. PKDD 2001: 42–53.
- [2] C. Chang, and C. Lin, *LIBSVM: a library for support vector machines*, Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, 2001.
- [3] W. Cheng, and E. Hüllermeier, *Combining instance-based learning and logistic regression for multilabel classification*, Mach. Learn., **76**:211–225, 2009.
- [4] K. Dembczynski, W. Cheng, and E. Hüllermeier, *Bayes Optimal Multilabel Classification via Probabilistic Classifier Chains*. ICML 2010: 279–286.
- [5] K. Dembczynski, W. Waegeman, W. Cheng, and E. Hüllermeier, *On label dependence in multi-label classification*, In ICML-MLD: 2nd International Workshop on learning from Multi-Label Data, 5–12, Haifa, Israel, 2010.
- [6] A. Dimou, G. Tsoumakas, V. Mezaris, I. Kompatsiaris, I. Vlahavas, *An Empirical Study Of Multi-Label Learning Methods For Video Annotation*, 7th International Workshop on Content-Based Multimedia Indexing, IEEE, Chania, Crete, 2009.
- [7] A. Genkin, D. D. Lewis, and D. Madigan, *Large-Scale Bayesian Logistic Regression for Text Categorization*, Technometrics. **49**(3), 291–304, 2007.
- [8] S. Godbole, and S. Sarawagi *Discriminative Methods for Multi-labeled Classification*. PAKDD 2004: 22–30.
- [9] W. R. Hersh, C. Buckley, T. J. Leone, and D. Hickman, *Ohsumed: an interactive retrieval evaluation and new large test collection for research*. Proc. of the ACM SIGIR Conference, 1994.
- [10] T. Joachims, *Text categorization with support vector machines: learning with many relevant features*. LS8-Report 23, Universität Dortmund, LS VIII-Report, 1997.
- [11] D. D. Lewis, and M. Ringuette, *A Comparison of Two Learning Algorithms for Text Categorization*, In: Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, 81–93, 1994.
- [12] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, *RCV1: A New Benchmark Collection for Text Categorization Research*. J. Mach. Learn. Res. **5**, 361–397, 2004.
- [13] X. Luo, and A. Nur Zincir-Heywood, *Evaluation of Two Systems on Multi-class Multi-label Document Classification*, ISMIS 2005: 161–169.
- [14] A. McCallum, and K. Nigam, *A Comparison of event models for Naive Bayes text classification*. Proc. of the AAAI-98 Workshop on Learning for Text Categorization, 1998.
- [15] A. McCallum, *Multi-label text classification with a mixture model trained by EM*, Proc. of the AAAI’ 99 Workshop on Text Learning, 1999.

- [16] G. Pandey, V. Kumar, and M. Steinbach, *Computational Approaches for Protein Function Prediction: A Survey*, TR 06-028, Department of Computer Science and Engineering, University of Minnesota, Twin Cities, 2006.
- [17] J. Platt, *Probabilistic outputs for Support Vector Machines and comparisons to regularized likelihood methods*, in *Advances in Large Margin Classifiers*, 61–74, MIT Press, 1999.
- [18] M. F. Porter, Snowball Package, available at <http://snowball.tartarus.org>.
- [19] A. E. Romero, *DauroLab*, Software available at <http://sourceforge.net/projects/daurolab>, 2010.
- [20] G. Salton, and M. E. Lesk, *The SMART automatic document retrieval systems—an illustration*. Commun. ACM **8**(6), 391–398, 1965.
- [21] F. Sebastiani, *Machine learning in automated text categorization*, ACM Comput. Surv. **34**(1), 1–47, 2002.
- [22] K. Trohidis, and G. Tsoumakas, G. Kalliris, and I. Vlahavas, *Multilabel Classification of Music into Emotions*, Proc. 9th International Conference on Music Information Retrieval (ISMIR 2008). 325–330, Philadelphia, PA, USA, 2008.
- [23] G. Tsoumakas, and I. Katakis, *Multi-Label Classification: An Overview*, Int. J. Data Warehouse. Min., **3**(3):1–13, 2007.
- [24] N. Ueda, and K. Saito, *Parametric Mixture Models for Multi-Labeled Text.*, NIPS 2002: 721–728.
- [25] T. Vander Wal, *Folksonomy Coinage and Definition*, available at <http://vanderwal.net/folksonomy.html>.
- [26] I. H. Witten, and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition, Morgan Kaufmann, 2005.
- [27] Y. Yang, and X. Liu, *A Re-Examination of Text Categorization Methods*. SIGIR 1999: 42–49.
- [28] Y. Yang, *A study of thresholding strategies for text categorization*, In *Proceedings of the SIGIR’01, 24th annual international ACM conference on Research and Development in Information Retrieval*, 137–145, 2001.
- [29] M. L. Zhang, and Z. H. Zhou, *A k-Nearest Neighbor Based Algorithm for Multi-label Classification*, Proc. of the 1st IEEE International Conference on Granular Computing, 2005.